Amendments to the Specification

Please replace the Abstract with the following amended paragraph:

A system and a corresponding method for monitoring and controlling server nodes in a clustered environment. The cluster includes a first instance and a second instance, each of the first and second instances including a number of server nodes. Each instance executes a control logic to start the instance by initiating a launch logic for each of the server nodes in the cluster. When initiated, the launch logic is configured to load a virtual machine in each respective server node and execute ~~Java~~JAVA processes in the virtual machine. The system further includes a communication interface coupled between the launch logic and the control logic to enable exchange of information between the ~~Java~~JAVA processes and the control logic. The launch logic is configured to obtain information regarding a status of each of the ~~Java~~JAVA processes running in the virtual machine and enable the control logic to access the status information via the communication interface.

Please replace paragraph [0003] with the following amended paragraph:

[0003] In accordance with one embodiment of the invention, a system and a corresponding method are disclosed for monitoring and controlling server nodes contained within a clustered environment. The cluster includes a first instance and a second instance, each of the first and second instances including a number of server nodes. The system includes a control logic to, among other things, start the cluster by initiating a launch logic for each of the server nodes contained in the first and second instances. When initiated, the launch logic is configured to load a virtual machine in each respective server node and execute ~~Java~~JAVA processes in the virtual machine. The system further includes a communication interface coupled between the launch logic and the control logic to enable exchange of information between the ~~Java~~JAVA processes and the control logic. The launch logic is configured to obtain information regarding a status of each of the ~~Java~~JAVA processes running in the virtual machine and enable the control logic to access the status information via the communication interface.

Please replace paragraph [0013] with the following amended paragraph:

[0013] FIG. 2 shows internal components of instances 112-1 and 112-2 and a central node 116 according to one embodiment of the invention. The first instance 112-1 includes a number of server nodes 201-1 through 201-N and a local dispatcher node 204, which receives requests from client computers and dispatches the requests across the multiple server nodes 202. Similarly, the second instance 112-2 includes a number of server nodes 210-1 through 210-N and a local dispatcher node 242, which dispatches requests from client computers across the multiple server nodes 210. In one embodiment, the server nodes implement a ~~Java~~JAVA platform, such as a ~~Java~~JAVA 2 Platform Enterprise Edition ("J2EE"). A J2EE is a ~~Java~~JAVA platform that can be used to develop, deploy and maintain enterprise applications.

Please replace paragraph [0015] with the following amended paragraph:

[0015] FIG. 3 illustrates an instance runtime environment 300 for monitoring and controlling of processes (e.g., ~~Java~~JAVA processes) executed within an instance of a clustered system according to one embodiment of the invention. Control logic 305 starts an instance by initiating a launch logic 335-1 through 335-N for each server node contained within the instance. In one embodiment, the control logic 305 and the launch logic 335 are responsible for monitoring and managing the status (e.g., starting, terminating and restarting) of the ~~Java~~JAVA processes running within the instance. In one embodiment, an instance is installed and runs on one hardware system. There may be one or more processes, which belong to the instance. Processes such as dispatcher and server nodes are started by the launch logic 335.

Please replace paragraph [0016] with the following amended paragraph:

[0016] The launch logic 335 is responsible for starting a ~~Java~~JAVA program or a set of ~~Java~~JAVA programs. In one embodiment, this is accomplished by loading a ~~Java~~JAVA virtual machine 357 and executing the processes in the ~~Java~~JAVA virtual machine. The executed processes may be used to provide services to client devices. The programs created using ~~Java~~JAVA may attain portability through the use of a ~~Java~~JAVA virtual machine. A ~~Java~~JAVA virtual machine is a software layer that provides an interface between compiled ~~Java~~JAVA binary code and the hardware platform which actually performs the program instructions. In one embodiment, the launch logic 335 is also configured to support the execution of standalone ~~Java~~JAVA programs, which may be executed without loading a ~~Java~~JAVA virtual machine.

Please replace paragraph [0017] with the following amended paragraph:

[0017] To enable the control logic 305 to communicate with the launch logic 335-1 through 335-N, a communication interface mechanism 325 is set up during the start up of the clustered system. The communication interface mechanism 325 may be based on any suitable communication mechanism such as shared memory, pipes, queues, signals, etc. In one embodiment, the shared memory 325 having a number of entries 330-1 through 330-N is set up to provide a way to exchange information between the ~~Java~~JAVA processes 355-1 through 355-N initiated by the launch logic 335 and the control logic 305. In one embodiment, the shared memory 325 is used to store information relating to the status of processes 355-1 through 355-N executed by the server nodes.

Please replace paragraph [0018] with the following amended paragraph:

[0018] The launch logic 335 is configured to obtain information regarding a status of each of the ~~Java~~JAVA processes 355 running in the virtual machine 357 and enable the control logic 305 to access the status information via the shared memory 325. In one embodiment, the status information is obtained by using a ~~Java~~JAVA native interface (JNI) 350 implemented within the launch logic 335. The JNI 350 enables the ~~Java~~JAVA processes 355 running inside the ~~Java~~JAVA virtual machine 357 to execute ~~Java~~JAVA native functionalities that are written and compiled for the underlying hardware platform. The ~~Java~~JAVA native processes are created using ~~Java~~JAVA native programming language, such as C and C++. During runtime, the JNI 350 is invoked by the launch logic 335 to communicate with the ~~Java~~JAVA processes 355 to update the shared memory 325 with information relating to the status of the ~~Java~~JAVA processes 355 running inside the ~~Java~~JAVA virtual machine.

Please replace paragraph [0020] with the following amended paragraph:

[0020] Based on the status information, the control logic 305 may control the operations of processes running within the instance by sending instructions to the launch logic 335 to, for example, start, terminate or restart a particular process. In one embodiment, the instructions to initiate the starting, terminating or restarting of a process or a server node is communicated via a suitable communication interface, such as named pipes 345-1 through 345-N. In response to the instructions received via the named pipes 345, the launch logic 335 controls the operation of the

corresponding process running in the virtual machine 357 via the ~~Java~~JAVA native interface 350.

Please replace paragraph [0024] with the following amended paragraph:

[0024] Referring to FIG. 4, the general operations involved in monitoring and controlling of ~~Java~~JAVA processes in a clustered system according to one embodiment of the invention are shown. In block 410, a control logic 305 executing within each instance reads a list of cluster elements (e.g., server nodes) to be started during start up of the clustered system. The list of server nodes to be started may be obtained from a configuration data 270 stored in a storage system 120 (shown in FIG. 2). Then in block 420, the control logic 305 initiates a launch logic 335 for each of the server nodes contained within the corresponding instance. When the launch logic 335 is launched on a particular server node, the launch logic 335 starts a process of constructing an appropriate run-time environment for ~~Java~~JAVA processes by loading a ~~Java~~JAVA virtual machine 357 and starting the specified processes in the virtual machine, in block 430.

Please replace paragraph [0025] with the following amended paragraph:

[0025] At the same time, the control logic 305 sets up a shared memory 325 to provide a means for exchanging information between the ~~Java~~JAVA processes initiated by the launch logic 335 and the control logic 305 in block 440. As indicated above, the shared memory 325 is used to store information relating to the status of processes executing within an instance. The status information stored in the shared memory 325 is periodically updated by the launch logic 335 in block 450. By accessing the updated information contained in the shared memory 325, this enables the control logic 305 to monitor the status of ~~Java~~JAVA processes in the instance, in block 460. Then, in block 470, the control logic 305 sends instructions to a launch logic 335 via a named pipe 345 to start, terminate or restart a particular process running in one of the server nodes.